

## A tener en cuenta

- No se corregirá código javascript en ningún archivo html.
- La única modificación admitida en los archivos html será la declaración del script correspondiente.
- Puesta en marcha de la API: `npm install y npm start`
- Puesta en marcha del proyecto Vite: `npm install y npm run dev`

Distribución por Resultados de Aprendizaje

- RA6 (DOM y estructura): 10 puntos
- RA7 (Comunicación asíncrona y almacenamiento): 10 puntos

## HARRY POTTER: *PERSONAJES*

La aplicación permite a los usuarios visualizar una lista de personajes de la saga Harry Potter en un formato de tarjetas y añadir nuevos personajes a través de un formulario. A continuación, se describen las funcionalidades y los archivos proporcionados:

### API JSON-mock *Personajes*:

Se proporciona una API simulada que devuelve un conjunto de personajes de Harry Potter en formato JSON. Esta API permite realizar las siguientes operaciones:

- lectura de todos los personajes
- lectura de un personaje
- añadir un personaje
- eliminar un personaje

Formato Datos: cada personaje tiene los siguientes atributos:

- **id**: identificador único del personaje.
- **nombre**: nombre del personaje.
- **casa**: la casa a la que pertenece.
- **patronus**: el patronus del personaje.
- **fechaNacimiento**: fecha de nacimiento del personaje.
- **sangre**: estado de sangre del personaje.
- **varita**: descripción de su varita.

### Formulario de Añadir Personaje:

Se proporciona el archivo `add-person.html` que contiene un formulario que permite a los usuarios añadir nuevos personajes. El formulario

contiene los siguientes campos

- **nombre:** campo de texto para ingresar el nombre del personaje.
- **casa:** campo desplegable para indicar entre las distintas casas de Hogwarts, la del personaje.
- **patronus:** campo desplegable para indicar el patronus del personaje entre distintos *patronus*.
- **fechaNacimiento:** campo de tipo fecha para ingresar la fecha de nacimiento del personaje.
- **sangre:** campo desplegable donde el usuario podrá elegir los diferentes tipos de sangre (pura, mestiza, hijo de muggles) del personaje.
- **varita:** campo de texto para describir la varita del personaje.

## Alta de Personajes

**Nombre: \***

**Casa: \***

Selecciona una casa

**Patronus: \***

Selecciona un Patronus

**Fecha de Nacimiento: \***

dd / mm / aaaa

**Estado de Sangre: \***

Selecciona el estado de sangre

**Varita:**

Dar de Alta

Los campos marcados con \* son obligatorios.

*Todos los campos son obligatorios, menos el campo de texto Varita. Sobre el formulario no se ha de realizar ninguna validación.*

### Listado de Personajes:

Se proporciona el archivo `index.html` que contiene un contenedor identificado como `personsContainer` donde se mostrará el listado de personajes, como se muestra a continuación:



Cada personaje se presentará en una tarjeta con su nombre, casa, patronus, fecha de nacimiento, estado de sangre y varita; por lo que, el usuario podrá visualizar todos los personajes disponibles. También contará con la opción de eliminar cualquier personaje de la lista.

### A REALIZAR ...

Implementa las siguientes acciones sobre el proyecto de Vite proporcionado:

1. El formulario enviará los datos a la API JSON-mock Personajes, para que el nuevo personaje sea añadido a la lista. Para realizar el envío solo se podrá utilizar la API Fetch.  
RA 7 - (2,5 puntos)
2. Cada vez que se añada un nuevo personaje, el nombre y la casa del mismo quedarán almacenados localmente en el navegador de manera permanente; siendo la clave el *nombre del personaje* y el valor la *casa del personaje*:  
RA 7 - (0,75 puntos)
3. Cuando un personaje es añadido o cuando clicas en Inicio, se redirigirá al usuario a la página de inicio (punto de entrada a la app); mostrándose el listado de personajes de la API JSON-mock Personajes. Para obtener los datos de los personajes solo se podrá utilizar la API Fetch.  
RA 7 - (2,5 puntos)

4. El archivo `index.html` contiene la *plantilla simplificada* de visualización de cada personaje. Muestra los mismos siguiendo este formato de tarjeta:  
RA 6 - (5 puntos)



5. Cuando se pulsa sobre el botón Eliminar, el personaje será borrado de la API JSON-mock Personajes; así como de la visualización. Para eliminar los datos de los personajes solo se podrá utilizar *XMLHttpRequest (XHR)*.  
RA 7 - (2,5 puntos)
6. Cuando se elimina el personaje de la API JSON-mock Personajes, también habrá que eliminar los datos almacenados localmente en el navegador para ese personaje.  
RA 7 - (0,75 puntos)
7. Manejo de errores en las peticiones realizadas con Fetch.  
En caso de error o respuesta no válida, deberá mostrarse un mensaje informativo en la interfaz.  
RA 7 – (1 punto)
8. Estructura del código.  
RA 6 - (5 puntos)

El código deberá estar organizado en varios módulos diferenciando, al menos:

- Llamadas a la API
- Manipulación del DOM
- Lógica principal de la aplicación